WEB INTELLIGENCE AND BIG DATA LAB

Name: Jaskirat Singh

Roll number: 05413202717

Section: CSE-1 (8th Semester)

EXPERIMENT-1

AIM: Create the data frame in big data and implement the following:

CODE:

```
import pandas as pd
weather_data = {
    'day': ['1/1/2017','1/2/2017','1/3/2017','1/4/2017','1/5/2017','1/6/2017'],
    'temperature': [32,35,28,24,32,31],
    'windspeed': [6,7,2,7,4,2],
    'event': ['Rain', 'Sunny', 'Snow','Snow','Rain', 'Sunny']
}
df = pd.DataFrame(weather_data)
df = pd.read_csv("weather_data.csv")
df
```

OUTPUT:

Out[3]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

```
In [4]: df.shape # rows, columns = df.shape
Out[4]: (6, 4)
```

In [5]:
```
df.head() # df.head(3)
```

Out[5]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain

In [6]:
```
df.tail() # df.tail(2)
```

Out[6]:

	day	temperature	windspeed	event
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

In [8]:
```
df.columns
```

Out[8]:
```
Index(['day', 'temperature', 'windspeed', 'event'], dtype='object')
```

In [9]:
```
df['day'] # or df.day
```

Out[9]:
```
0    1/1/2017
1    1/2/2017
2    1/3/2017
3    1/4/2017
4    1/5/2017
5    1/6/2017
Name: day, dtype: object
```

In [10]:
```
type(df['day'])
```

Out[10]:
```
pandas.core.series.Series
```

In [11]:
```
df[['day','temperature']]
```

Out[11]:

	day	temperature
0	1/1/2017	32
1	1/2/2017	35
2	1/3/2017	28
3	1/4/2017	24
4	1/5/2017	32
5	1/6/2017	31

In [12]:
```
df['temperature'].max()
```

Out[12]: 35

In [13]:
```
df[df['temperature']>32]
```

Out[13]:

	day	temperature	windspeed	event
1	1/2/2017	35	7	Sunny

In [14]:
```
df['day'][df['temperature'] == df['temperature'].max()] # Kinda doing SQL in pandas
```

Out[14]:
```
1    1/2/2017
Name: day, dtype: object
```

In [15]:
```
df[df['temperature'] == df['temperature'].max()] # Kinda doing SQL in pandas
```

Out[15]:

	day	temperature	windspeed	event
1	1/2/2017	35	7	Sunny

In [16]:
```
df['temperature'].std()
```

Out[16]: 3.8297084310253524

In [17]:
```
df['event'].max() # But mean() won't work since data type is string
```

Out[17]:
```
'Sunny'
```

In [18]:
```
df.describe()
```

Out[18]:

	temperature	windspeed
count	6.000000	6.000000
mean	30.333333	4.666667
std	3.829708	2.338090
min	24.000000	2.000000
25%	28.750000	2.500000
50%	31.500000	5.000000
75%	32.000000	6.750000
max	35.000000	7.000000

EXPERIMENT-2

AIM: Write different ways of creating pandas data frame at least 5.

CODE:

In [14]:
```
df = pd.read_csv("weather_data.csv")
df
```

Out[14]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow

In [24]:
```
df=pd.read_excel("weather_data.xlsx","Sheet1")
df
```

Out[24]:

	day	temperature	windspeed	event
0	2017-01-01	32	6	Rain
1	2017-01-02	35	7	Sunny
2	2017-01-03	28	2	Snow

In [37]:
```
import pandas as pd
weather_data = {
    'day': ['1/1/2017','1/2/2017','1/3/2017'],
    'temperature': [32,35,28],
    'windspeed': [6,7,2],
    'event': ['Rain', 'Sunny', 'Snow']
}
df = pd.DataFrame(weather_data)
df
```

Out[37]:

	day	event	temperature	windspeed
0	1/1/2017	Rain	32	6
1	1/2/2017	Sunny	35	7
2	1/3/2017	Snow	28	2

In [3]:
```
weather_data = [
    ('1/1/2017',32,6,'Rain'),
    ('1/2/2017',35,7,'Sunny'),
    ('1/3/2017',28,2,'Snow')
]
df = pd.DataFrame(data=weather_data, columns=['day','temperature','windspeed','event'])
df
```

Out[3]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow

In [4]:
```
weather_data = [
    {'day': '1/1/2017', 'temperature': 32, 'windspeed': 6, 'event': 'Rain'},
    {'day': '1/2/2017', 'temperature': 35, 'windspeed': 7, 'event': 'Sunny'},
    {'day': '1/3/2017', 'temperature': 28, 'windspeed': 2, 'event': 'Snow'},

]
df = pd.DataFrame(data=weather_data, columns=['day','temperature','windspeed','event'])
df
```

Out[4]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow

EXPERIMENT-3

AIM: How to handle missing data in pandas using fillna, interpolate and dropna methods.

CODE:

In [26]:
```
import pandas as pd
df = pd.read_csv("weather_data.csv",parse_dates=['day'])
type(df.day[0])
df
```

Out[26]:

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
1	2017-01-04	NaN	7.0	Sunny
2	2017-01-05	28.0	NaN	Snow
3	2017-01-06	NaN	7.0	NaN
4	2017-01-07	32.0	NaN	Rain
5	2017-01-08	NaN	NaN	Sunny
6	2017-01-09	NaN	NaN	NaN
7	2017-01-10	34.0	8.0	Cloudy
8	2017-01-11	40.0	12.0	Sunny

In [27]:
```
df.set_index('day',inplace=True)
df
```

Out[27]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	NaN	7.0	Sunny
2017-01-05	28.0	NaN	Snow
2017-01-06	NaN	7.0	NaN
2017-01-07	32.0	NaN	Rain
2017-01-08	NaN	NaN	Sunny
2017-01-09	NaN	NaN	NaN
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [28]:
```
new_df = df.fillna(0)
new_df
```

Out[28]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	0.0	7.0	Sunny
2017-01-05	28.0	0.0	Snow
2017-01-06	0.0	7.0	0
2017-01-07	32.0	0.0	Rain
2017-01-08	0.0	0.0	Sunny
2017-01-09	0.0	0.0	0
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [29]:
```
new_df = df.fillna({
        'temperature': 0,
        'windspeed': 0,
        'event': 'No Event'
    })
new_df
```

Out[29]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	0.0	7.0	Sunny
2017-01-05	28.0	0.0	Snow
2017-01-06	0.0	7.0	No Event
2017-01-07	32.0	0.0	Rain
2017-01-08	0.0	0.0	Sunny
2017-01-09	0.0	0.0	No Event
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

EXPERIMENT-4

AIM: How to handle the ffill ,bfill, interpolate().

CODE:

```
In [60]: new_df = df.fillna(method="ffill")
         new_df
```

Out[60]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	32.0	7.0	Sunny
2017-01-05	28.0	7.0	Snow
2017-01-06	28.0	7.0	Snow
2017-01-07	32.0	7.0	Rain
2017-01-08	32.0	7.0	Sunny
2017-01-09	32.0	7.0	Sunny
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

```
new_df = df.fillna(method="bfill")
new_df
```

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	28.0	7.0	Sunny
2017-01-05	28.0	7.0	Snow
2017-01-06	32.0	7.0	Rain
2017-01-07	32.0	8.0	Rain
2017-01-08	34.0	8.0	Sunny
2017-01-09	34.0	8.0	Cloudy
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

```
new_df = df.fillna(method="bfill", axis="columns") # axis is either "index" or "columns"
new_df
```

	temperature	windspeed	event
day			
2017-01-01	32	6	Rain
2017-01-04	7	7	Sunny
2017-01-05	28	Snow	Snow
2017-01-06	7	7	NaN
2017-01-07	32	Rain	Rain
2017-01-08	Sunny	Sunny	Sunny
2017-01-09	NaN	NaN	NaN
2017-01-10	34	8	Cloudy
2017-01-11	40	12	Sunny

```
new_df = df.fillna(method="ffill",limit=1)
new_df
```

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	32.0	7.0	Sunny
2017-01-05	28.0	7.0	Snow
2017-01-06	28.0	7.0	Snow
2017-01-07	32.0	7.0	Rain
2017-01-08	32.0	NaN	Sunny
2017-01-09	NaN	NaN	Sunny
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

```
new_df = df.interpolate()
new_df
```

	temperature	windspeed	event
day			
2017-01-01	32.000000	6.00	Rain
2017-01-04	30.000000	7.00	Sunny
2017-01-05	28.000000	7.00	Snow
2017-01-06	30.000000	7.00	NaN
2017-01-07	32.000000	7.25	Rain
2017-01-08	32.666667	7.50	Sunny
2017-01-09	33.333333	7.75	NaN
2017-01-10	34.000000	8.00	Cloudy
2017-01-11	40.000000	12.00	Sunny

EXPERIMENT-5

AIM: How dataframe.replace method can be used to replace specific values with some other values.

CODE:

Replacing single value

```
new_df = df.replace(-99999, value=np.NaN)
new_df
```

	day	temperature	windspeed	event
0	1/1/2017	32 F	6 mph	Rain
1	1/2/2017	-99999	7 mph	Sunny
2	1/3/2017	28 c	-99999	Snow
3	1/4/2017	-99999	7	0
4	1/5/2017	32	-99999	Rain
5	1/6/2017	31	2	Sunny
6	1/6/2017	34	5	0

Replacing list with single value

```
new_df = df.replace(to_replace=[-99999,-88888], value=0)
new_df
```

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	0.0	7.0	Sunny
2017-01-05	28.0	0.0	Snow
2017-01-06	0.0	7.0	0
2017-01-07	32.0	0.0	Rain
2017-01-08	31.0	2.0	Sunny

Replacing per column

```
new_df = df.replace({
        'temperature': -99999,
        'windspeed': -99999,
        'event': '0'
    }, np.nan)
new_df
```

	day	temperature	windspeed	event
0	1/1/2017	32.0	6.0	Rain
1	1/2/2017	NaN	7.0	Sunny
2	1/3/2017	28.0	NaN	Snow
3	1/4/2017	NaN	7.0	NaN
4	1/5/2017	32.0	NaN	Rain
5	1/6/2017	31.0	2.0	Sunny
6	1/6/2017	34.0	5.0	NaN

EXPERIMENT-6

AIM: How dataframe.replace method can be used In mapping replace specific values with some other values.

CODE:

Replacing by using mapping

```
new_df = df.replace({
        -99999: np.nan,
        'no event': 'Sunny',
    })
new_df
```

	day	temperature	windspeed	event
0	1/1/2017	32.0	6.0	Rain
1	1/2/2017	NaN	7.0	Sunny
2	1/3/2017	28.0	NaN	Snow
3	1/4/2017	NaN	7.0	0
4	1/5/2017	32.0	NaN	Rain
5	1/6/2017	31.0	2.0	Sunny
6	1/6/2017	34.0	5.0	0

Regex

```
# when windspeed is 6 mph, 7 mph etc. & temperature is 32 F, 28 F etc.
new_df = df.replace({'temperature': '[A-Za-z]', 'windspeed': '[a-z]'},'', regex=True)
new_df
```

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	-99999	7	Sunny
2	1/3/2017	28	-99999	Snow
3	1/4/2017	-99999	7	0
4	1/5/2017	32	-99999	Rain
5	1/6/2017	31	2	Sunny
6	1/6/2017	34	5	0

Replacing list with another list

```
df = pd.DataFrame({
    'score': ['exceptional','average', 'good', 'poor', 'average', 'exceptional'],
    'student': ['rob', 'maya', 'parthiv', 'tom', 'julian', 'erica']
})
df
```

	score	student
0	exceptional	rob
1	average	maya
2	good	parthiv
3	poor	tom
4	average	julian
5	exceptional	erica

```
df.replace(['poor', 'average', 'good', 'exceptional'], [1,2,3,4])
```

	score	student
0	4	rob
1	2	maya
2	3	parthiv
3	1	tom
4	2	julian
5	4	erica

EXPERIMENT-7

AIM: How to use pandas concat function to join or append dataframes In big data analysis.

CODE:

```
us_weather = pd.DataFrame({
    "city": ["new york","chicago","orlando"],
    "temperature": [21,14,35],
    "humidity": [68, 65, 75]
})
us_weather
```

	city	humidity	temperature
0	new york	68	21
1	chicago	65	14
2	orlando	75	35

```
df = pd.concat([india_weather, us_weather])
df
```

	city	humidity	temperature
0	mumbai	80	32
1	delhi	60	45
2	banglore	78	30
0	new york	68	21
1	chicago	65	14
2	orlando	75	35

```
df = pd.concat([india_weather, us_weather], ignore_index=True)
df
```

	city	humidity	temperature
0	mumbai	80	32
1	delhi	60	45
2	banglore	78	30
3	new york	68	21
4	chicago	65	14
5	orlando	75	35

.

```
df = pd.concat([india_weather, us_weather], keys=["india", "us"])
df
```

		city	humidity	temperature
india	**0**	mumbai	80	32
	1	delhi	60	45
	2	banglore	78	30
us	**0**	new york	68	21
	1	chicago	65	14
	2	orlando	75	35

```
df.loc["us"]
```

	city	humidity	temperature
0	new york	68	21
1	chicago	65	14
2	orlando	75	35

```
df.loc["india"]
```

	city	humidity	temperature
0	mumbai	80	32
1	delhi	60	45
2	banglore	78	30

EXPERIMENT-8

AIM: Implement Merge Data frames in big data analysis.

CODE:

```
import pandas as pd
df1 = pd.DataFrame({
    "city": ["new york","chicago","orlando"],
    "temperature": [21,14,35],
})
df1
```

	city	temperature
0	new york	21
1	chicago	14
2	orlando	35

```
df2 = pd.DataFrame({
    "city": ["chicago","new york","orlando"],
    "humidity": [65,68,75],
})
df2
```

	city	humidity
0	chicago	65
1	new york	68
2	orlando	75

```
df3 = pd.merge(df1, df2, on="city")
df3
```

	city	temperature	humidity
0	new york	21	68
1	chicago	14	65
2	orlando	35	75

```
df1 = pd.DataFrame({
    "city": ["new york","chicago","orlando", "baltimore"],
    "temperature": [21,14,35, 38],
})
df1
```

	city	temperature
0	new york	21
1	chicago	14
2	orlando	35
3	baltimore	38

```
df2 = pd.DataFrame({
    "city": ["chicago","new york","san diego"],
    "humidity": [65,68,71],
})
df2
```

	city	humidity
0	chicago	65
1	new york	68
2	san diego	71

```
df3=pd.merge(df1,df2,on="city",how="inner")
df3
```

	city	temperature	humidity
0	new york	21	68
1	chicago	14	65

```
df3=pd.merge(df1,df2,on="city",how="outer")
df3
```

	city	temperature	humidity
0	new york	21.0	68.0
1	chicago	14.0	65.0
2	orlando	35.0	NaN
3	baltimore	38.0	NaN
4	san diego	NaN	71.0

```
df3=pd.merge(df1,df2,on="city",how="left")
df3
```

	city	temperature	humidity
0	new york	21	68.0
1	chicago	14	65.0
2	orlando	35	NaN
3	baltimore	38	NaN

```
df3=pd.merge(df1,df2,on="city",how="right")
df3
```

	city	temperature	humidity
0	new york	21.0	68
1	chicago	14.0	65
2	san diego	NaN	71

```
df3=pd.merge(df1,df2,on="city",how="outer",indicator=True)
df3
```

	city	temperature	humidity	_merge
0	new york	21.0	68.0	both
1	chicago	14.0	65.0	both
2	orlando	35.0	NaN	left_only
3	baltimore	38.0	NaN	left_only
4	san diego	NaN	71.0	right_only

EXPERIMENT-9

AIM: What is pivot? Use pivot() function and its arguments , What is a pivot table? Use pivot_table() in dataframe.

CODE:

```
import pandas as pd
import numpy as np
df = pd.read_csv("weather.csv")
df
```

	date	city	temperature	humidity
0	5/1/2017	new york	65	56
1	5/2/2017	new york	66	58
2	5/3/2017	new york	68	60
3	5/1/2017	mumbai	75	80
4	5/2/2017	mumbai	78	83
5	5/3/2017	mumbai	82	85
6	5/1/2017	beijing	80	26
7	5/2/2017	beijing	77	30
8	5/3/2017	beijing	79	35

```
df.pivot(index='city',columns='date')
```

	temperature			humidity		
date	**5/1/2017**	**5/2/2017**	**5/3/2017**	**5/1/2017**	**5/2/2017**	**5/3/2017**
city						
beijing	80	77	79	26	30	35
mumbai	75	78	82	80	83	85
new york	65	66	68	56	58	60

```
df.pivot(index='city',columns='date',values="humidity")
```

date	**5/1/2017**	**5/2/2017**	**5/3/2017**
city			
beijing	26	30	35
mumbai	80	83	85
new york	56	58	60

```
df.pivot(index='date',columns='city')
```

	temperature			humidity		
city	**beijing**	**mumbai**	**new york**	**beijing**	**mumbai**	**new york**
date						
5/1/2017	80	75	65	26	80	56
5/2/2017	77	78	66	30	83	58
5/3/2017	79	82	68	35	85	60

```
df.pivot(index='humidity',columns='city')
```

	date			temperature		
city	**beijing**	**mumbai**	**new york**	**beijing**	**mumbai**	**new york**
humidity						
26	5/1/2017	None	None	80.0	NaN	NaN
30	5/2/2017	None	None	77.0	NaN	NaN
35	5/3/2017	None	None	79.0	NaN	NaN
56	None	None	5/1/2017	NaN	NaN	65.0
58	None	None	5/2/2017	NaN	NaN	66.0
60	None	None	5/3/2017	NaN	NaN	68.0
80	None	5/1/2017	None	NaN	75.0	NaN
83	None	5/2/2017	None	NaN	78.0	NaN
85	None	5/3/2017	None	NaN	82.0	NaN

Pivot Table

```
df = pd.read_csv("weather2.csv")
df
```

	date	city	temperature	humidity
0	5/1/2017	new york	65	56
1	5/1/2017	new york	61	54
2	5/2/2017	new york	70	60
3	5/2/2017	new york	72	62
4	5/1/2017	mumbai	75	80
5	5/1/2017	mumbai	78	83
6	5/2/2017	mumbai	82	85
7	5/2/2017	mumbai	80	26

EXPERIMENT-10

AIM: Implement the Reshape function in big data analysis.

CODE:

```
import pandas as pd
df = pd.read_csv("weather.csv")
df
```

	day	chicago	chennai	berlin
0	Monday	32	75	41
1	Tuesday	30	77	43
2	Wednesday	28	75	45
3	Thursday	22	82	38
4	Friday	30	83	30
5	Saturday	20	81	45
6	Sunday	25	77	47

```
melted = pd.melt(df, id_vars=["day"], var_name='city', value_name='temperature')
melted
```

	day	city	temperature
0	Monday	chicago	32
1	Tuesday	chicago	30
2	Wednesday	chicago	28
3	Thursday	chicago	22
4	Friday	chicago	30
5	Saturday	chicago	20
6	Sunday	chicago	25
7	Monday	chennai	75
8	Tuesday	chennai	77
9	Wednesday	chennai	75
10	Thursday	chennai	82
11	Friday	chennai	83
12	Saturday	chennai	81
13	Sunday	chennai	77
14	Monday	berlin	41